“ Grab ’em by the Fallacy ”
Stance Detection to Identify Fake News

CS585, UMass Amherst, Fall 2017

Ajinkya Zadbuke **Arhum Savera**

azadbuke@cs.umass.edu asavera@umass.edu

Abstract

The Fake News Challenge was organized as a response to the recent emergence and increase of fake news in journalism and social media. In our project we aim to implement a solution for this challenge by detecting stance in news articles. We explore multiple machine learning models, employing natural language processing to construct features from the data. Our best model, the Averaged Bag-of-Words MLP, beats the baseline model provided by the challenge.

1 Introduction

With the constant deluge of information available through multiple sources, it is more important than ever to be able to distinguish between factual and fabricated reports. Fake news, defined by the New York Times as ”a made up story with an intention to deceive” has been widely cited as a contributor to the outcome of the 2016 US elections. The Fake News Challenge (FNC-1) was introduced in 2016 to explore how machine learning and natural language processing technologies could be used to tackle the problem of Fake News.

Stance detection is the process of comparing articles and headlines from multiple sources and attempting to detect conflicts in perspective relative to the topic. It can be a helpful first step towards building AI-assisted systems to detect Fake News, particularly in light of clickbaity, intentionally misleading and fallacious reporting that is prevalent today.

We focus on stance detection for the above, rather than explicitly modelling the fake news problem. Truth labelling is a challenging task even for humans, and other tasks have been determined to be feasible but decidedly non-trivial [2]. For our project, we try to detect the stance of a news article based on its body text and relative to the headline. Such a system would assist human fact-checkers in gathering multiple sources that agree, disagree or discuss the input claim or headline. Table 1 shows a sample article from the dataset with its associated stances.

Robert Plant Ripped up $800M Led Zeppelin Reunion Contract	Stance
Led Zeppelin's Robert Plant turned down £500 MILLION to reform supergroup ...	Agree
No, Robert Plant did not rip up an $800 million deal to get Led Zeppelin...	Disagree
Robert Plant reportedly tore up an $800 million Led Zeppelin reunion deal...	Discuss
Richard Branson's Virgin Galactic is set to launch SpaceShipTwo today...	Unrelated

Table 1: Example of data - article body and stances

2 Related Work

Ferreira and Vlachos introduce the Emergent data set that we will be using in our project [2]. The authors also present their approach for stance detection where a regularized logistic regression classifier is used and features are extracted from the headline and the associated claim. The performance of the model is comparable to the state of the art stance detection. This team summarized each article into a headline and used a logistic regression model with features representing the article and claim to classify the combination of article and claim as either "for," "against," or "observing," with a final accuracy level of 73%.

Davis and Proctor work on the same fake news dataset that we are using, implementing a Concatenated Multi-Layer Perceptron model, with Bag of Words representations and GLoVE Embeddings [3]. They have also tried RNN and LSTM models, with the MLP performing better than the others.

Conroy et al. also tackle the problem of fake news detection but use a combination of linguistic cue approaches (with machine learning) and network analysis approaches [4]. They experimented with BoW, Probabilistic Context-Free Grammars and SVMs while taking Social Network behavior and linked data properties into account.

Augenstein et al. experiment with Conditional LSTM encoding in context of a tweet-to-target model, with a slightly different labelling scheme [5]. It involves encoding the target as a fixed-length vector, and then the tweet with state intialized as the target's repesentation. The final output of the tweet LSTM is used to predict the stance label for the pair.

Rocktaschel et al. reported that using conditional encoding of LSTM models coupled with neural attention mechanisms have impressive results in detecting textual entailment [6]. This task is conceptually similar to stance detection. Two sentences are input and the task is to determine if the sentences are unrelated, contradict each other, or if the second sentence is a logical consequence of the first sentence.

Before stance detection, closely related research, Natural language inference, focused on analyzing the relationship between two short sentences. MacCartney explores a range of approaches to the problem of natural language inference beginning with methods which are robust but approximate, and proceeding to progressively more precise approaches, and attempts to infer if a given hypothesis can be inferred from a given premise [7].

Mohammad et al. focused on stance detection in tweets and tried to analyze the relationship between tweets and their views on specific topics [8]. Given a tweet and a target entity, they determine whether the tweeter is in favor of the given target, against the given target, or whether neither inference is likely. They used SVM-ngrams and other simple linear classifiers using BOW for stance detection.

Bowman et al. also tackle the problem of natural language inference, wherein the authors introduced a new corpus of sentence pairs labeled for entailment, contradiction, and independence [9]. They found that simple lexicalized models and neural network models perform considerably well.

Faulkner attempts stance classification for annotated student essays [10]. Using a novel set of linguistically motivated features including Part-of-speech generalized dependency subtrees and polarity scores, multinomial Naive Bayes and SVMs (RBF kernel) showed significant increases in accuracy relative set baselines.

3 Datasets

The dataset we will use is the one provided for the original FNC-1 task [2]. This is a collection of articles, each structured as (headline, body, stance). The stance, which is our label, in this case, is one of {unrelated, disagree, agree, discuss}. The dataset has been created and annotated by accredited journalists, so we are not concerned with verifying quality of the data. We have 1648 headlines and 1669 article bodies, which have paired to create 49972 body-headline pairs. The data is inherently imbalanced as a result of the pairwise combinations, such that around 73% of the examples are labeled as 'unrelated', with the rest distributed between 'discuss', 'agree' and 'disagree' (see Table 2). The below figures show length of the article bodies and headlines in tokens. Table 2 shows the train-test split selected for training and evaluating our models. We use 10-fold cross-validation for selecting the best performing model, and a separate holdout (dev) set for evaluating accuracy. The test set gives the final score for the model and was the primary evaluation metric for FNC-1.

Label	**Stance**	**Ratio**
0	Agree	0.0736
1	Disagree	0.0168
2	Discuss	0.1782
3	Unrelated	0.7313

Table 2: Data Distribution

Train	**Dev**	**Test**
35687+4663	9622	25413

Table 3: Train-Test Split

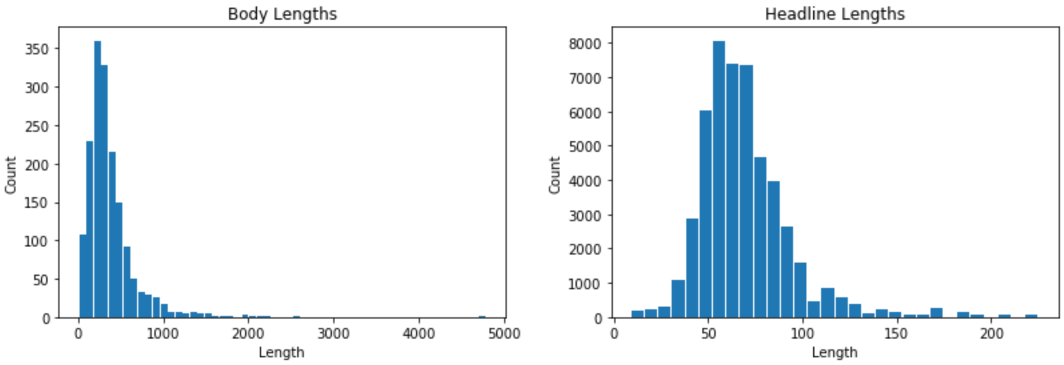

Figure 1: Dataset token lengths

4 Evaluation

Keeping with the spirit of the challenge, we will follow their evaluation guidelines. This involves handling the class imbalance by assigning a 25-75 weighting for the 'unrelated' and other classes, respectively (Figure 2). This scheme rewards correct classification of the stance over relatedness, as the latter is much easier to solve. The baseline accuracy provided by the challenge serves as a good benchmark to judge our models against (currently at 79.53 on the dev set and 75.20 on the test set).

5 Experiments

For preprocessing, we use NLTK to tokenize the article and headline bodies, and normalize them to lowercase. We also perform removal of punctuation and stopwords.

5.1 Features

We use the following features as input to our models. Feature engineering is an important component of the machine learning pipeline, and contributes significantly to performance of the model.

5.1.1 Word overlapping features

The first feature is the jaccard similarity between the headline and the body of the article. After tokenizing the headlines and bodies, we calculate the Jaccard similarity for each headline and body pair.

5.1.2 Refuting features

We use a defined list of refuting words such as 'fake','fraud','hoax','false' etc. and return a binary vector for each headline with each dimension being 1 if the corresponding refuting word is present in the headline.

5.1.3 Polarity features

In this context, we define the polarity of a string to be 1 if it contains an odd number of refuting terms or 0 if it contains an even number of refuting terms. We use the same list of refuting words described above. We also define a polarity pair to be the polarity of a headline and body of an ariticle. We iterate over the articles and return a polarity pair for each article.

5.1.4 N-Gram hits

In our experiments, we use 2, 3, 4, 5 and 6 grams. We compute the n-grams of the headline and then count how many times they appear in the body of the article. Additionally, we keep a separate count of how many common n-grams appear in the first 256 characters of the article body.

5.1.5 Char-grams hits

In our experiments, we use 2, 4, 8 and 16 character-grams. We compute the character-grams of the headline and then count how many times they appear in the body of the article. For these too, we keep a separate count of how many common character-grams appear in the first 256 characters of the article body.

5.1.6 Co-occurence count

We tokenize the headline and body and count how many times a token in the headline occurs in the article body. We also keep a separate count for how many times common tokens appear in the first 256 characters of the article body.

5.1.7 Co-occurence count without stop words

We tokenize the headline and body, remove stop words, and count how many times a token in the headline occurs in the article body. We also keep a separate count for how many times common tokens appear in the first 256 characters of the article body.

5.1.8 GloVe Averaged Vectors

To incorporate information about the semantic content of the given article or headline, we use pre-trained GloVE (Global Vectors for Word Representation) embeddings. The version we use are trained on the Wikipedia and GigaWord corpora, with 400k types and 200 dimensions for each word [11]. We average the vectors for the first 200 words of the text (both body and headline, separately), normalize and concatenate them to form the feature vector. If we encounter an out-of-vocabulary word, we intialize a random vector for that word and include it among the others.

5.1.9 Word Count Vectors

We also include additional information about the dataset by constructing a simple Bag of Words model. We take the top 1000 words, sorted by frequency of occurrence in the training data, and build the feature vector depending on whether the word occurs in the given text and if so, its count.

5.2 Models

Hyperparameters for the models given below are shown in Table 4. Each model is trained for 20 epochs.

5.2.1 Gradient Boosting

This is the baseline model used by FNC-1. Gradient Boosting is an ensemble model that combines many weak learners (Decision trees) into a single, strong learner over multiple iterations. It performs admirably over a limited feature set, and is a reasonably difficult baseline to beat. The model we use consists of 200 estimators.

5.2.2 Logistic Regression

Logistic Regression is a classification model. The input features are passed through a linear layer and then a sigmoid or logistic non-linearity for predicting the label, where:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

$$y = \sigma(Wx + b)$$

5.3 Multilayer Perceptron

This is a simple neural network, with 3 hidden layers and ReLU (rectified linear unit) activation functions. We use hidden layer sizes of [128, 64, 16].

$$relu(x) = max(0, x)$$

5.4 Average Bag-of-words MLP

This model is an extension on the previous one, with a larger number of parameters. We use Leaky ReLU as the non-linearity. In addition, we also use Dropout as a regularization mechanism, with a factor of 0.3 (this fraction of neurons is switched off during the forward pass at random). For both this and the previous model, we use the Adam optimizer, with a batch size of 100 and a learning rate of 0.001. Further, we initialize all weights using the Xavier initialization scheme [14].

$$leaky\_relu(x) = \text{if } x < 0, \text{ then } 0.01x, \text{ else } x$$

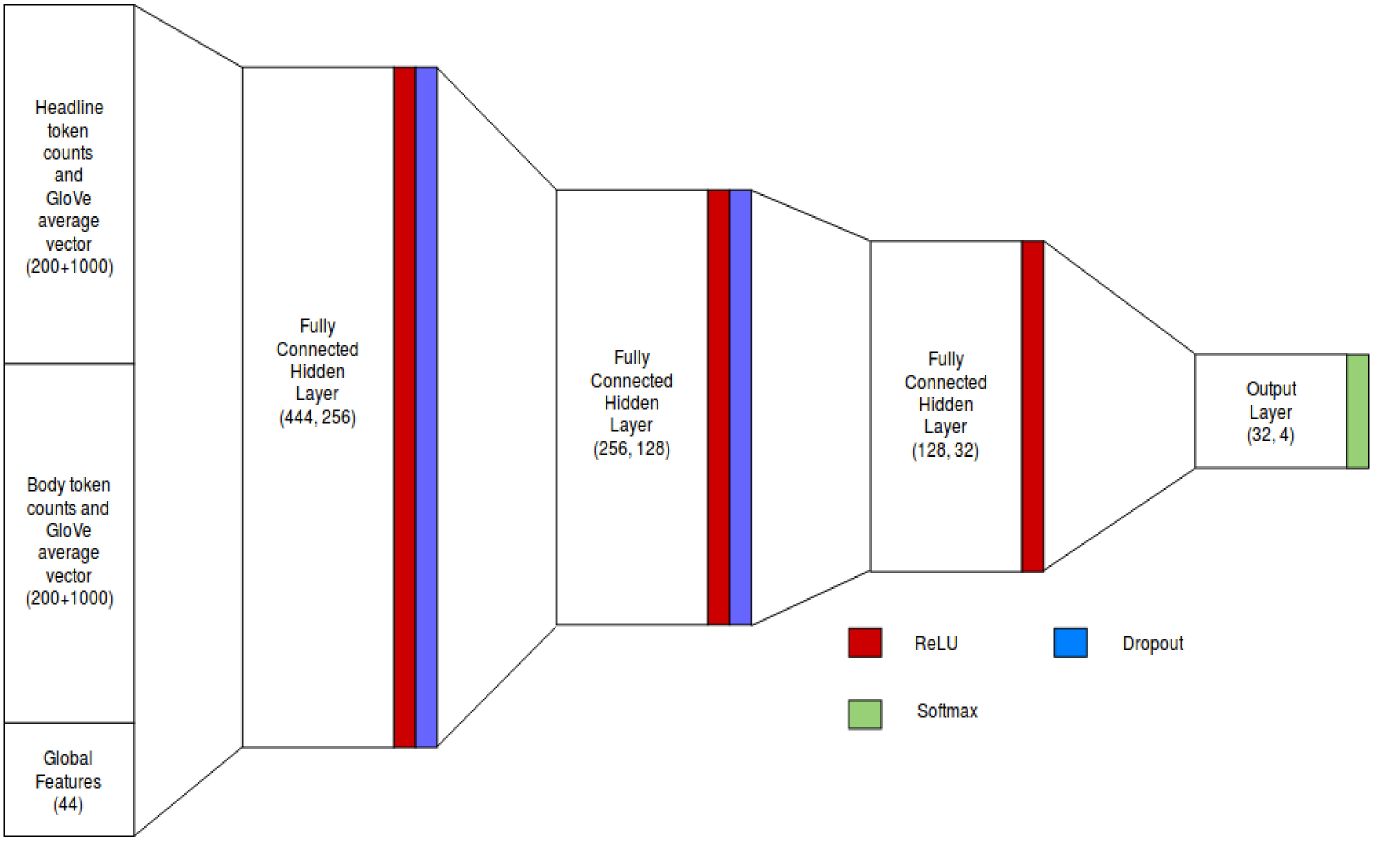
Headline token counts and GloVe average vector (200+1000)
Body token counts and GloVe average vector (200+1000)
Global Features (44)
Fully Connected Hidden Layer (444, 256)
Fully Connected Hidden Layer (256, 128)
Fully Connected Hidden Layer (128, 32)
Output Layer (32, 4)
ReLU
Dropout
Softmax

Figure 2: Average bag-of-words MLP

Feature			
GloVe dimensions	100	**200**	300
BoW Vocab	500	**1000**	2000
Batch Size	100	**200**	500
Learning Rate	**0.001**	0.005	0.01
Dropout	**0.3**	0.4	0.5
Hidden Layer Sizes	[512, 128, 32, 8]	**[256, 128, 32]**	[128, 64, 16]

Table 4: Hyper Parameter tuning

6 Results

Model	Dev Set	Test Set	Dev Macro F1	Test Macro F1
Gradient Boosting	79.53	75.20	0.499	0.456
Logistic Regression	76.138	72.11	0.458	0.43
Multilayer Perceptron	80.12	74.97	0.590	0.53
Averaged BoW MLP	**85.80**	**77.51**	**0.711**	**0.551**

Table 5: Results

The Gradient Boosting model performed quite well considering complexity relative to the neural models. Work done by Riedel et al. showed that using a straightforward bag of words MLP can perform competitively with more complex ensemble models [12]. Using one hidden layer of 100 units and a softmax on the output of the final linear layer, they outperformed models implemented by Pfohl et al. including LSTM with Conditional Encoding and Attention mechanisms [13]. Since our main objective was to achieve a higher score on the competition set, we focused more time on implementing a Bag-of-words MLP. Additionally, we did try to implement basic recurrent models (RNNs and LSTMs), but these proved difficult to train due to hardware constraints. Still, our BoW model matches expectations, with dev and test set accuracy scores of 85.8 and 77.5 respectively.

7 Conclusion and Future Work

In this report, we proposed and developed a solution for stance detection to identify fake news. Using a tuned Bag-of-words Neural Network with multiple layers and proper tuning, we were able to beat the baseline scores provided by FNC-1. The challenge was an interesting one, and we intend to work further on the same topic. Possible approaches we can try are Convolutional and Recurrent models, Bidirectional LSTMs, attention mechanisms etc. Such models might help to uncover additional dependencies between parts of the article, and provide better semantic understanding. Chaining multiple classifiers together, for example to classify relatedness and stance respectively can help in fine-grained distinction between the classes. We can also attempt to correct the class imbalance by augmenting the dataset with externally collected data.

8 References

[1] Fake News Challenge - https://www.fakenewschallenge.org

[2] Ferreira, W., and Vlachos, A., *Emergent : A Novel Dataset for Stance Classification*, (2016)

[3] Davis, R., Proctor, C., *Fake News, Real Consequences: Recruiting Neural Networks for the Fight Against Fake News*, (2017)

[4] Conroy, N., Rubin, V., Chen, Y. *Automatic Deception Detection: Methods for Finding Fake News*, (2015)

[5] Augenstein, I., Rocktaschel, T., Vlachos, A., Bontcheva, K., *Stance Detection with Conditional Bidirectional Encoding*, (2016)

[6] Rocktaschel, T., Grefenstette, E., Hermann, K.M., Kocisky, T., and Blunsom, P. *Reasoning about Entailment with Neural Attention*, (2015).

[7] MacCartney, B., *Natural language inference*, (2009)

[8] Mohammad, S.M., Kiritchenko, S., Sobhani, P., Zhu, X., Cherry, C., textitSemeval-2016 task 6: Detecting Stance in Tweets, (2016)

[9] Bowman, S.R., Angeli, G., Potts, C., and Manning, C.D, *A large annotated corpus for learning natural language inference*, (2015)

[10] Faulkner, A., *Automated Classification of Stance in Student Essays: An Approach Using Stance Target Information and the Wikipedia Link-Based Measure*, (2014)

[11] Pennington, J., Socher, R., and Manning, C.D., *GloVE:Global Vectors for Word Representation* (2014) [12] Riedel, B., Augenstein, I.,Spithourakis, G., Riedel, S., *A simple but tough-to-beat baseline for the Fake News Challenge stance detection task*, (2017)

[13] Pfohl, S., Triebe, O., Legros, F., *Stance Detection for the Fake News Challenge with Attention and Conditional Encoding*, (2017)

[14] Glorot, X., and Bengio, Y., *Understanding the difficulty of training deep feedforward neural networks*, (2010)